CMPE 221_223_242
Fall 2022
Programming Homework 1

This assignment is due by 23:59 on Sunday, 6 November 2022.

You are welcome to ask your HW related questions. You should use only one of these options:

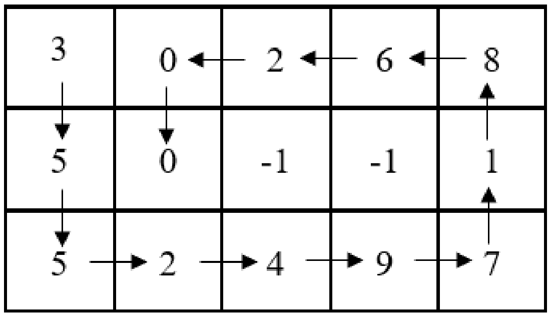
1. Moodle Homework **Question Forum**: HW Question-and-Answer (Q&A) Forum on Moodle is always available. Use the "Forum" link at the course Moodle page.
2. Homework **RECITATION HOURS**: There will be three Q&A RECITATION HOURs on the following days:

   CMPE223-HW1-OfficeHour1: 17 Oct, 18:00-20:00 PM, Zoom ID: 953 6825 8296
   CMPE223-HW1-OfficeHour2: 24 Oct, 18:00-20:00 PM, Zoom ID: 965 4830 2030
   CMPE223-HW1-OfficeHour3: 31 Oct, 18:00-20:00 PM, Zoom ID: 985 7599 5261

Note: Please make sure that you have read the HW document well before participating. However, no HW related questions will be accepted except from the above options.

PROGRAMMING TASK

Q1: You must use Linked List for solution of this question. Your code will be tested using .txt files, design your code considering this.



3	0	2	6	8
5	0	-1	-1	1
5	2	4	9	7

A. Return the elements of the given matrix in spiral form. The program should stop when it sees -1. The starting point will be 3 and the direction will be down until the end of the column, then right until the end of the row, up to the top and left.

The sequence should be as follows:

3→5→5→2→4→9→7→1→8→6→2→0→0

Example Input/Output:

```
Input filename:matrix.txt
[3, 5, 5, 2, 4, 9, 7, 1, 8, 6, 2, 0, 0]
```

B. The lists will be given as input, for example:

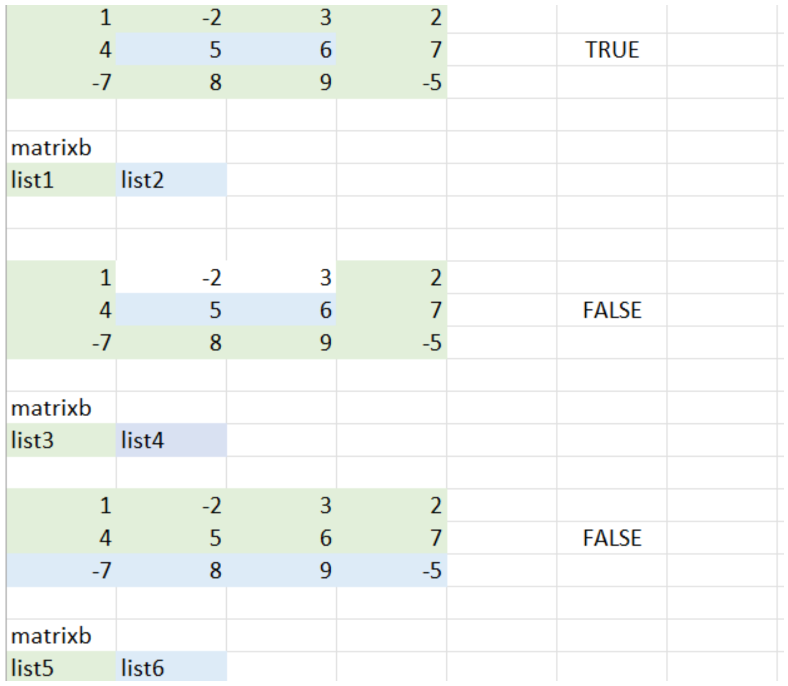
List 1: {3, 5, 5, 2, 4, 9, 7, 1, 8, 2, 6, 2, 0, 0}
List 2: {0, -1, -1}

You need to find out if the neighborhood of list 1 contains list 2 or not. To do this, you need to check whether any node from list 1 is connected to any node in list 2. The nodes to the right, left, under, and above should all be checked. (Checking diagonally is not necessary.)

Example Input/Output:

```
Input filename:matrixb.txt
Input filename:list1.txt
Input filename:list2.txt
TRUE
```

The direction of the arrow is the same as in 1A. You need to read 3 txt files, the first is the matrixb, the second and third are the lists with the indices of the matrices. The output will be TRUE or FALSE.



1	-2	3	2		
4	5	6	7		TRUE
-7	8	9	-5		
matrixb					
list1	list2				
1	-2	3	2		
4	5	6	7		FALSE
-7	8	9	-5		
matrixb					
list3	list4				
1	-2	3	2		
4	5	6	7		FALSE
-7	8	9	-5		
matrixb					
list5	list6				

You must use Linked List for solution of this question!

Q2: You must use stack/queue for the solution of this question! Your code will be tested using .txt files, design your code considering this.

Implement a Java solution that calculates the range of bitcoin values for the current day, given the series of n daily values of this coin.

Range definition:

The maximum number of consecutive days (starting today and going backwards) that the bitcoin value was higher than or equal to today's value.

For example, if the value of a bitcoin over the next 6 days were 90, 65, 70, 60, 75, 80, then the range returned would be 1, 2, 1, 4, 1, 1.

Example Input/Output:

```
Input filename:price2.txt
[90, 65, 70, 60, 75, 80]
[1, 2, 1, 4, 1, 1]
```

Explanation:

90 → return 1 because it is the first element of the range
65 → return 2 because 65 is smaller than 90
70 → return 1 because 90 is greater than 70
60 → return 4 because the last 4 values (60,70,65,90) were higher than or equal to today's value
75 → return 1 because 90 is greater than 75
80 → return 1

You must use stack/queue for the solution of this question!

WHAT TO HAND IN

A zip file for both parts containing:

- The Java sources for your program.
- The Java sources should be WELL DOCUMENTED as comments, as part of your grade will be based on the level of your comments.
- You should test your Java source files on (if) available Moodle VPL environment to ensure your code solution's correctness before submitting. VPL simply tests your program's output by checking against given sample input. You should pass that task's VPL test case successfully.

- A **maximum-3 pages** PDF report document that explains your own answers for programming task in a clearly readable PA report format (refer to **PA REPORT FORMAT** section).
- For given task, only code or report submission will not be graded. In other words, you should submit both correct code solution and its related report for the task in order to be graded.

A programming assignment report is a self-description of a programming assignment and your solution. The report must not be hand-written. You may use a word processor or the on-line editor of your choice and prepare as a PDF document. The report must be grammatically correct and use complete English sentences. Each report should include the following sections, in the order given:

Information (%2.5): This section includes your ID, name, section, assignment number information properly.

Problem Statement and Code Design (%15): Include a brief summary of the problem and/or your sub-tasks to be completed in this assignment. You should show your modular design rationale by creating a structure chart that indicates your top-down, stepwise refinement of the problem solution. You may create the structure chart using available graphical tools like MS PowerPoint, SmartDraw etc.

Implementation, Functionality (%20): Since you have modular source code, you should describe each sub-module (program) in this section. Each sub-module should include names and types of any input/output parameters as well as the pseudocode algorithm that used for completing its task. By this way, you give meaning to each chart boxes from the previous section.

Testing (%7.5): You should provide a tester class that is able to identify key test points of your program. This class should be able to generate additional (apart from the given sample input/output) test data for the purpose of being clear on what aspects of the solution are being tested with each set. This section should also include a description of any program *bugs* that is, tests which has incorrect results. You should write these to describe your tests, summarize your results, and argue that they cover all types of program behavior.

Final Assessments (%5): In this final section, you should briefly answer the following questions:

- What were the trouble points in completing this assignment?
- Which parts were the most challenging for you?
- What did you like about the assignment? What did you learn from it?

GRADING:

- Codes (%50, Q1:25, Q2:25)
  - Available test cases evaluation on VPL: %15, (Q1:7.5, Q2:7.5)
  - Hidden test cases evaluation: %15, (Q1:7.5, Q2:7.5)
  - Approach to the problem: %20, (Q1:10, Q2:10)

- Report (%50)
  - Information: %2.5
  - Problem Statement and Code design: %15
  - Implementation, Functionality: %20
  - Testing: %7.5
  - Final Assessments: %5

Submitting report without codes will not be evaluated!!

IMPORTANT

IMPORTANT NOTES: Do not start your homework before reading these notes!!!

1. This assignment is due by 23:59 on Thursday, October 27th.

2. You should upload your homework to Moodle before the deadline. No hardcopy submission is needed. You should upload files and any additional files if you wrote additional classes in your solution as a single archive file (e.g., zip, rar).

3. The standard rules about late homework submissions apply (20 points will be deducted for each late day). Please see the course syllabus for further discussion of the late homework policy as well as academic integrity.

4. You ARE NOT ALLOWED to modify the given method names. However, if necessary, you may define additional data members and member functions.

5. Your classes' name MUST BE as shown in the homework description.

6. The submissions that do not obey these rules will not be graded.
7. To increase the efficiency of the grading process as well as the readability of your code, you must follow the following instructions about the format and general layout of your program.

8. Do not forget to write down your id, name, section, assignment number or any other information relevant to your program in the beginning of your Java files. Example:

```
//-----------------------------------------------------
// Title: Scheduler tester class
// Author: Name/Surname
// ID: 2100000000
// Section: 1
// Assignment: 1
// Description: This class tests the …
//-----------------------------------------------------
```

9. Since your codes will be checked without your observation, you should report everything about your implementation. Add detailed comments to your classes, functions, declarations

etc. Make sure that you explain each function in the beginning of your function structure. Example:

```
void setVariable(char varName, int varValue)
//-----------------------------------------------------------
// Summary: Assigns a value to the variable whose
// name is given.
// Precondition: varName is a char and varValue is an
// integer
// Postcondition: The value of the variable is set.
//-----------------------------------------------------------
{
      // Body of the function
}
```

10. Indentation, indentation, indentation...

11. This homework will be graded by your TA, Merve Işıl Peten. Thus, you may ask her your homework related questions through HW forum on Moodle course page.